AIM: Implementation of Logistic Regression.

```
import pandas as pd
# used to read the data set
import·numpy·as·np
#·used·to·do·some·operations·with·the·arrays
import·os
# used handle some files
import matplotlib.pyplot as plt
# used to visualize the data using graphs
import seaborn as sns
# plotting the chart in a single line
```

```
df = pd.read_csv("/content/Iris.csv")
```

```
df.head()
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Id             150 non-null    int64  
 1   SepalLengthCm  150 non-null    float64
 2   SepalWidthCm   150 non-null    float64
 3   PetalLengthCm  150 non-null    float64
 4   PetalWidthCm   150 non-null    float64
 5   Species        150 non-null    object 
dtypes: float64(4), int64(1), object(1)
memory usage: 7.2+ KB
```

```
df = df.drop(columns = ['Id'])
df.head(5)
```

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

```
df['Species'].value_counts()
```

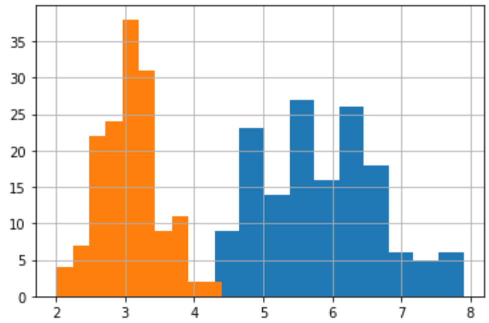
```
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
Name: Species, dtype: int64
```

```
df.isnull().sum()
```

```
SepalLengthCm    0
SepalWidthCm     0
PetalLengthCm    0
PetalWidthCm     0
Species          0
dtype: int64
```

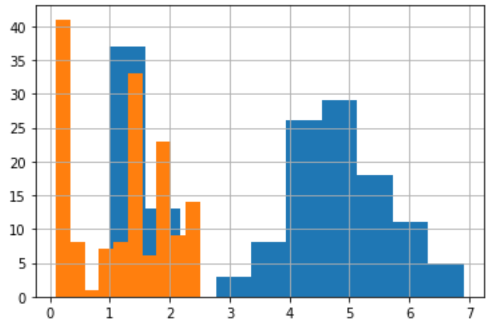
```
df['SepalLengthCm'].hist()
df['SepalWidthCm'].hist()
```

```
<AxesSubplot:>
```



```
df['PetalLengthCm'].hist()
df['PetalWidthCm'].hist()
```

```
<AxesSubplot:>
```



```
df.corr()
```

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
SepalLengthCm	1.000000	-0.109369	0.871754	0.817954
SepalWidthCm	-0.109369	1.000000	-0.420516	-0.356544
PetalLengthCm	0.871754	-0.420516	1.000000	0.962757
PetalWidthCm	0.817954	-0.356544	0.962757	1.000000

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
df['Species'] = le.fit_transform(df['Species'])
df.head(100)
```

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0
...	...	...	...	...	...
95	5.7	3.0	4.2	1.2	1
96	5.7	2.9	4.2	1.3	1
97	6.2	2.9	4.3	1.3	1
98	5.1	2.5	3.0	1.1	1
99	5.7	2.8	4.1	1.3	1

100 rows × 5 columns

```
from sklearn.model_selection import train_test_split
```

```
X = df.drop(columns = ['Species'])
Y = df['Species']
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.25)
```

```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
```

```
model.fit(X_train, Y_train)
```

```
▾ LogisticRegression
LogisticRegression()
```

```
print("Accuracy: ", model.score(X_test, Y_test) * 100)
```

```
Accuracy:  97.36842105263158
```

```
classifier = LogisticRegression(random_state = 0, solver='lbfgs', multi_class='auto')
classifier.fit(X_train, Y_train)
```

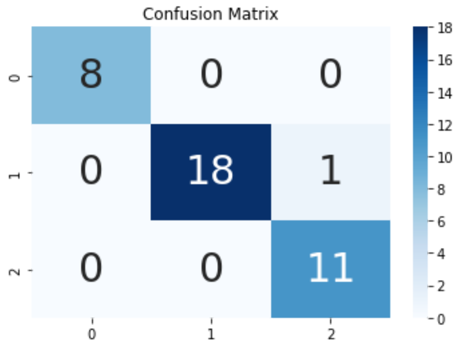
```
▾        LogisticRegression
LogisticRegression(random_state=0)
```

```
# Predicting the Test set results
y_pred = classifier.predict(X_test)
```

```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(Y_test, y_pred)
print(cm)
```

```
[[ 8  0  0]
 [ 0 18  1]
 [ 0  0 11]]
```

```
# Plot confusion matrix
import seaborn as sns
import pandas as pd
# confusion matrix sns heatmap
ax = plt.axes()
df_cm = cm
sns.heatmap(df_cm, annot=True, annot_kws={"size": 30}, fmt='d',cmap="Blues", ax = ax )
ax.set_title('Confusion Matrix')
plt.show()
```



```
from sklearn.metrics import mean_squared_error
from math import sqrt
print("MSE Score: ", mean_squared_error(y_pred, Y_test))
print("RMSE Score: ", np.sqrt(mean_squared_error(y_pred, Y_test)))
```

```
MSE Score:  0.02631578947368421
RMSE Score:  0.16222142113076254
```

✓ 0s completed at 2:33 PM ● ×